Искаков С.Н., студент
Рогович В.И., проф., канд. физ.-мат. наук

РАЗВИТИЕ ПАРАДИГМЫ ВЗАИМОДЕЙСТВИЯ

В связи с тем, что все современные машины построены по принципу машины тьюринга, то, следовательно, класс задач, решаемых вычислительными машинами, ограничен теми задачами, которые алгоритмически разрешимы. В связи с этим возникает проблема взаимодействия вычислительной машины с окружающей средой. Это связано с тем, что алгоритм обладает свойствами: детерминированность, дискретность, элементарность, результативность.

Чтобы выйти из этого затруднения, следует выделить несколько основных моментов, которые должны быть соблюдены в новой парадигме построения вычислительных машин: а) количество агентов, взаимодействующих с ВМ, в общем случае не должно быть конечно; б) ВМ машина должна быть открытой системой, и, следовательно, все ресурсы не должны разделяться на локальные и сетевые; в) остальные ВМ, участвующие во взаимодействии, являются внешними агентами; г) ВМ должна быть более выразительна, чем машина тьюринга (ТМ), то есть решать больший круг задач, чем ТМ, включая алгоритмические.

В парадигме взаимодействия основную роль играет взаимодействие между объектами, и, как следствие, большую роль играет объектно-ориентированное программирование. При попытке смоделировать алгоритмическими методами (которые являются основными) взаимодействие между объектами может получиться, что данная задача моделирования не решаема данным инструментарием, так как зачастую объект не может быть описан алгоритмически из-за спонтанных и необъяснённых взаимодействий этого объекта с другими объектами и с окружающей средой.

Машины тьюринга: ТМ – машины с изменяющимся состоянием $M = (S, T, s0, F)$, с конечными наборами состояний S, лентой символов T, стартовым состоянием $s0$, и оператором перехода $F: SxT \to SxO$. ТМ преобразовывает конечный вход строк $x \in T^*$ к выходам $y = M(x)$ конечной последовательностью шагов. На каждом шаге ТМ читает символ ленты i, исполняет переход состояния (s, i)-> (s', o), пишет символ o, и/или перемещает считывающую головку на одно положение вправо или влево. ТМ вычисляют функции $f: X \to Y$, переводящие целые числа в целые, строки в строки. Класс функций, вычисляемых ТМ, называют эффективно-вычислимыми функциями.

Существует два наиболее распространенных подхода для решения парадигмы взаимодействия. К ним относится *SIM* - sequential interaction machine (последовательно взаимодействующая машина) и *MIM- MultiStream Interaction Machine* (многопотоковая машина взаимодействия). Рассмотрим оба подхода подробней и сравним их.

Sequential Interaction Machine (SIM): *SIMs* - машины с изменяющимся состоянием $M = (S, I, m)$, где S - счетный набор состояний, I - счетный набор динамически связанных входов, m – оператор перехода $m: SxI \to SxO$ переводящий пары состояния-действия в новые состояния и выходы.

Каждый шаг вычисления *SIM ((s, i)-> (s', o))* может рассматриваться как полное вычисление ТМ, где *i* - динамически подаваемый входной символ (строка), выход *o* может повлиять на последующие входы, и *s'* - следующее состояние SIM. Элементы и *S* и *I* конечны в любое данное время, но их размер неограничен. Поведение SIMs выражено **потоками ввода - вывода: потоки ввода - вывода** имеют форму *(i1, o1), (i2, o2)...*, где *ok* вычислено от *ik*, но предшествует *ik+1* и может влиять на него. Зависимость *ik+1* от *ok* называют **сцеплением ввода - вывода (input-output coupling)**.

Сцепление ввода - вывода нарушает разделение областей и диапазонов, порождая динамическую зависимость входов от предшествующих выходов, которая является характерной для взаимодействующего вопроса-ответа, диалога, игр с двумя людьми, и процессов контроля. Переходы *m* от *sk* до *sk+1* пар ввода - вывода связаны с парами ввода - вывода *(ik, ok)*.

Persistent Turing Machines (PTMs) - каноническая модель для SIMs, которая минимально расширяет ТМ.

Persistent Turing Machines (PTMs): PTM - MultiTape TM с постоянной рабочей лентой, содержание которой сохраняется между взаимодействиями.

Состояние PTM хранится на его не стираемой рабочей ленте и может быть бесконечно перечислимо. Отдельные взаимодействия PTM соответствуют вычислениям TM.

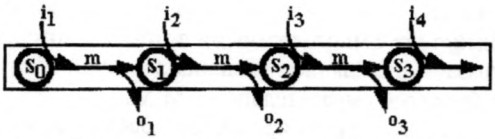

Рис. 1. SIM с входами *i∈I, выходами o∈O и переходом m*

MultiStream Interactive Machine (MIM) - конечные агенты, которые взаимодействуют с многократными автономными потоками: например, распределенные базы данных, которые предоставляют услуги множеству автономных клиентов. Поведение MIM невыразимо через SIM в том смысле, что MIM могут выполнить больший класс задач, чем SIMs. Это контрастирует с тем фактом, что ТМ с несколькими рабочими лентами не более выразительный, чем ТМ с одной рабочей лентой. ТМ выражает поведение единственного агента, SIM выражает взаимодействие 2 агентов, в то время как MIM выражают взаимодействие *n* агентов для *n>2*. Анализ большей выразительности взаимодействия с 3 агентами, чем взаимодействие с 2 агентами, обеспечивает вычислительную структуру для того, чтобы показать, что проблема с 3 телами в физике не сводится к проблеме с 2 телами. MIM обеспечивают точное определение **распределенных систем**: система (вычислительный агент) является распределённой, если ее взаимодействия могут быть описаны MIM, но не SIM. Это определение аккуратно определяет распределенное поведение для единственного конечного вычислительного агента в терминах параллелизма (nonserializability) его диалогового поведения. Понятие, что "распределенный = параллельно взаимодействующий", является удобным из-за его точности и простоты. Его центральным понятием является параллелизм взаимодействия в противоположность распре-

деленности данных. Это может сначала казаться нестандартным, но фактически выражает распределенность в терминах естественного понятия нелокальности агентов. MIM, которые не могут быть описаны как SIM, выражают свойство нелокальности агентов, которые взаимодействуют с MIM. Предложение, что MIM более выразительны, чем SIM, подразумевает, что распределённые системы (определенные выше) более выразительны, чем нераспределенные системы. Окружающие среды MIM взаимодействуют одновременно с конечными агентами. Взаимодействующий параллелизм отличается качественно от параллельного выполнения действий и является формой "параллелизма второго порядка". Параллельные системы со многими параллельными процессами могут быть невзаимодействующими (закрыты), взаимодействующими с единственным внешним потоком или взаимодействующими с многократными потоками. Распределенное вычисление может быть точно определено как параллельное взаимодействие: распределенные системы имеют параллелизм второго порядка, в то время как нераспределенные системы взаимодействуют самое большее с одним потоком. Большая выразительность машины M1, чем M2, для окружающей среды многократного потока E может в принципе быть определена свойством, которое показывает, что отношение эквивалентности наблюдения EQ(M1, E) лучше, чем EQ(M2, E), так же, как для SIMs. Однако, эквивалентности, вызванные MIM на взаимодействии, исчисляемости, и тезисе Чёрча 6/20 части окружающей среды, сложнее, чем эквивалентности, вызванные SIM, включая параллельное взаимодействие с многократными потоками, которые не могут быть непосредственно связаны с созданием внешних различий. Ниже приведены два вида аргументов за большую выразительность MIM:

MIM могут иметь недетерминированное поведение, не выразимое SIM. MIM моделируют распределенные системы, которые не могут быть полностью рассмотрены никаким последовательным наблюдателем (потоком). Каждый поток MIM моделирует наблюдателя, который рассматривает MIM через специфический интерфейс, но нет никакого наблюдателя, который может наблюдать MIM целиком.

Системы, состоящие из SIM с наблюдателем, закрыты, в то время как MIM с любым первичным наблюдателем открыты, так как они могут быть косвенно изменены вторичными наблюдателями (см. рис.2). Модификация MIM вторичными наблюдателями воспринята первичным наблюдателем как недетерминированное поведение.

MIM, которые являются детерминированными с точки зрения metaobserver, наблюдающего поведение всей системы в целом, кажутся недетерминированными с точки зрения любого отдельного первичного наблюдателя с ограниченными последовательными полномочиями наблюдения.

MIM могут иметь nonserializable поведение, не выразимое SIMs. Nonserializability автономных потоков обеспечивает более полное определение большей выразительности MIM, чем недетерминизм. Многократные автономные потоки не могут быть преобразованы в один поток путём последовательного их разбиения, потому что валентность сцепления потоков ввода - вывода не может быть сохранена при слиянии для задач, которые требуют взаимодействия меж-

ду потоками. Особенно простая форма координации: nonserializability для отправки подзадач.

Если *(iu, ou)* - взаимодействие ввода - вывода агента *A* с пользователем *U* и *(oe, ie)* – взаимодействие посредством *A* с экспертом *E*, к кому *A* отправляет подзадачу в ответе на запросы *U*, то взаимодействия происходят во временной последовательности *(iu, oe, ie, ou)*. Пользовательский элемент потока *(iu, ou)* и опытное взаимодействие потока *(oe, ie)* нельзя рассмотреть как раздельное и независимое. Нет никакого последовательного порядка, который достигает желательного эффекта при выполнении этих двух взаимодействий *(iu, ou)* и *(oe, ie)*. Этот пример показывает, что даже простые формы координации подобно отправке не могут быть выражены SIM из-за нарушения serializability.

Координация нарушает serializability (валентность) сцепления потоков ввода – вывода.

Добавление автономных потоков (каналов наблюдения) к взаимодействующему вычислительному агенту увеличивает его выразительность, тогда как добавление неинтерактивных лент к ТМ просто увеличивает структурную сложность предопределенных входов и не затрагивает выразительную мощь. Ограничение MIM к serializable поведению, так же как SIMs к неинтерактивному поведению, это способ, который может быть трактован в пользу решения проблемы.

MIM более выразительны, чем SIMs, в то время как multitape ТМ и singletape ТМ одинаково выразительны.

Преобразование в последовательную форму эффекта многократных потоков ограничивает мощности конечных агентов, решающих проблему, так как nonserializable поведение компьютеров и людей более важно, чем вообще понимается. Nonserializable поведение - особенность сотрудничества, которая отличает менеджеров высокого уровня, взаимодействующих с множеством подчиненными, от рабочих сборочной линии, которые взаимодействуют с единственным потоком. MIM моделируют важные формы высокоуровневого человеческого поведения. Способность характеризовать совместное поведение MIM и различать совместное от последовательно выраженного может оказаться важной в формализации совместного вычисления.

Сотрудничество, координация и управление смоделированы MIM, но не SIMs или PTMs.

MIM поддерживают поведение nonserializable взаимодействия и истинного параллелизма, в то время как SIMs поддерживает только serializable параллелизм чередования и сделки. Мы догадываемся, что более богатое поведение nonserializability и истинного параллелизма может в принципе быть выражено формой эквивалентности наблюдения, которая обобщает bisimulation. Абстракция от ТМ до рекурсивно счетных множеств сравнима с абстракцией от SIMs до non-well-founded множеств, которые моделируют поведение единственных потоков. Существует предположение, что MIM могут аналогично быть согласованно определены аксиомами теории множеств, точная форма которой требует дальнейшего исследования.

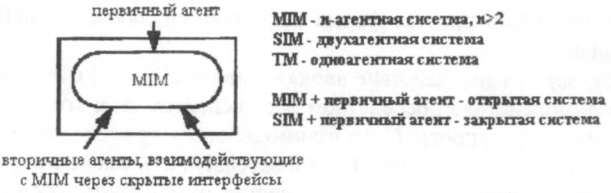

Рис.2. Многопотоковое взаимодействие первичного и вторичного агентов

Анализируя предложенные варианты, следует отметить, что SIM, моделируется последовательностными алгоритмами. MIM в отличие от SIM является более перспективным направлением для дальнейшего исследования.

Относительно MIM следует отметить, что он не практичен в том виде, в котором существует. Это обусловлено предположениями, что

1) количество агентов, взаимодействующих с MIM, ограничено;

2) существует первичный агент.

На наш взгляд, не должно существовать первичного или вторичного агента, с которым взаимодействует MIM. Все агенты должны быть первичными. Разграничение приоритетов между агентами должно осуществляться посредством внутреннего интерфейса MIM, который также должен осуществлять поддержку взаимодействия с внешней средой и создавать локальные интерфейсы для каждого агента. При этом количество создаваемых локальных интерфейсов в общем случае не должно быть ограниченным.

Если же предполагать, что количество агентов, взаимодействующих с MIM, ограничено, то, следовательно, можно выбрать предельное количество агентов и выстроить алгоритмическую модель системы на это количество агентов, что в свою очередь будет означать, что нет смысла строить какую-либо новую модель, в частности MIM.